VŠB - Technická univerzita Ostrava
Fakulta elektrotechniky a informatiky
Katedra informatiky

Absolvování individuální odborné praxe

Individual Professional Practise in the Company

2010

Petr Lukáš

Prohlašuji, že jsem tuto bakalářskou práci vypracoval samostatně. Uvedl jsem všechny literární prameny a publikace, ze kterých jsem čerpal.

V Ostravě 27. dubna 2010 ……..

Chtěl bych poděkovat vedení firmy za umožnění vykonávání bakalářské práce formou odborné praxe a zejména svému konzultantovi, paní Ing. Daně Kallerové, která mi svými obrovskými zkušenostmi v oblasti analýzy informačních systémů průběžně pomáhala s řešením zadaných úkolů.

Abstrakt

Práce popisuje průběh vykonávaní bakalářského semináře formou individuální odborné praxe ve firmě NATIVA Enterprise s.r.o. Popis jednotlivých zadaných úloh zahrnuje vždy význam úkolu, principy řešení a stručnou charakteristiku zákazníka, aby byl význam lépe pochopen. Závěrem práce je shrnuto, kterých okruhů znalostí nabytých během studia bylo při absolvování praxe využito a které okruhy bylo nutné dodatečně nastudovat.

Klíčová slova

odborná praxe, NATIVA Enterprise s.r.o., bakalářský seminář, informační systém, analýza, Huisman, AutoCont, databáze

Abstract

This bachelor thesis describes the course of the individual professional practise in the NATIVA Enterprise s.r.o. company. Each particular description includes the purpose of the task, the principles of the solution and a short information about the customer to clarify the purpose. The conclusion resumes what skills learnt during the study were used to solve the problems and what additional skills were needed to learn.

Keywords

professional practice, NATIVA Enterprise s.r.o., bachelor seminar, information system, analysis, Huisman, AutoCont, database

Seznam použitých zkratek

CAD	Computer-aided Design
CAM	Computer-aided Manufacturing
CNC	Computer Numerical Control
DTS	Data Transformation Services
DXF	Drawing Exchange Format
EAN	European Article Number
EXE	Executable file
HDW	Huisman Data Warehouse
LINQ	Language Integrated Query
MTO	Material Take-off List
NTFS	New Technology File System
PDA	Personal Digital Assistant
PO	Project Order
SP	Store Place
SŘBD	Systém řízení báze dat
SQL	Structured Query Language
T-SQL	Transact SQL
UML	Unified Modeling Language
WMS	Warehouse Management Systém
XML	Extensible Markup Language

Obsah

1 Úvod

Má spolupráce s firmou NATIVA Enterprise s.r.o. (dále jen Nativa) začala již v polovině roku 2007, tedy ještě před započetím studia na Vysoké škole báňské. Obsah této práce tedy bude vztahován nejen na období 5. a 6. semestru, ale na celou dobu mého působení ve firmě.

Během praxe jsem se setkal s velkým množstvím zajímavých problémů, při jejichž řešení bylo nutné jak aplikovat znalosti nabyté studiem, tak vyvíjet vlastní iniciativu při vzdělávání formou samostudia.

V této bakalářské zprávě bych chtěl popsat některé úkoly, jež byly z mého pohledu nebo z pohledu firmy nejpodstatnější. Cílem popisu jednotlivých úkolů bude především nastínit, které části byly z odborného hlediska nejproblematičtější, v čem problémy spočívaly a jaké postupy jsem volil při jejich řešení.

2 Popis pracovní činnosti a zaměření

Primární oblastí působení firmy Nativa je analýza a vývoj informačních systémů, vzdělávání v oblasti aplikačního software a poskytování technické podpory. Jedná se především o tvorbu systémů na míru podle potřeb konkrétních zákazníků [5].

Hlavní náplní mé odborné praxe v této firmě byla analýza a implementace informačních systémů, avšak nezřídka jsem se setkával také s vývojem speciálních grafických nebo výpočetních aplikací. Po technické stránce jsem pracoval převážně s databází Microsoft SQL Server 2005 a s platformou .NET Framework 2.0. Nejčastěji jsem používal programovací jazyky SQL na straně databázového serveru a Visual Basic .NET nebo Visual C# na straně desktopové aplikace.

Součástí vývoje byla také průběžná tvorba projektových dokumentací, které slouží mimo jiné jako podklady pro uživatelské příručky a nápovědy. Zde bylo nutné aplikovat znalosti anglického jazyka, jelikož v mnoha případech systémy slouží nebo budou sloužit v zahraničí.

Nemalou část praxe tvořila také komunikace se zákazníky. Bylo nutné pružně reagovat na jejich požadavky, vyhodnocovat, zda jsou požadavky smysluplné a odhadovat jejich časovou náročnost. Jak již bylo uvedeno, mnohdy se jednalo o zahraniční zákazníky, takže i zde byla nutná znalost angličtiny.

Během praxe jsem se podílel jak na vývoji nových systémů, kde jsem strukturu zdrojových kódů a databáze znal od začátku, tak na rozšiřování již existujících systémů, kde bylo nutné zorientovat se ve stávajícím stavu databázového schématu nebo zdrojového kódu.

3 Průběh odborné praxe

Jednotlivé úkoly jsem zde rozčlenil podle zákazníků, pro které výsledné produkty slouží nebo sloužit v nejbližší době budou. Popis úloh zahrnuje částečně také některé body analýzy, na které jsem se ve firmě rovněž podílel. Veškeré popsané úkoly jsem, pokud není uvedeno jinak, převážně implementoval sám.

3.1 Huisman

Jedním z nejdůležitějších zákazníků firmy Nativa je společnost Huisman. Aby byl význam jednotlivých úkolů lépe pochopen, rád bych na úvod této kapitoly stručně popsal infrastrukturu a zaměření této organizace.

Jedná se o celosvětově známou společnost působící v oblasti designu a výroby těžké techniky sloužící nejen na pevnině, ale také na moři. Společnost se zabývá především návrhem a výrobou ropných plošin, lodních a námořních jeřábů, mobilních jeřábů apod. Hlavní sídlo a jeden z výrobních závodů se nachází ve městě Schiedam v Nizozemí. Dva hlavní výrobní závody se nacházejí zde, v České republice, ve Sviadnově u Frýdku Místku, a v Číně v oblasti Xiamen, v provincii Fujian. Dvě další pobočky, které se zabývají především prodejem a designem, mají své sídlo ve Spojených státech v Houstonu a v Brazílii ve městě Rio de Janeiro [4].

Spolupráce firem Huisman a Nativa byla započata zde, v České republice, v pobočce Sviadnov. Na počátku mého externího působení byl firmou Nativa vyvinut informační systém sloužící speciálně pro potřeby zdejší pobočky Huisman. Informační systém pojmenovaný SysAdmin byl a je zaměřen především na evidování projektů, možnosti jejich podrobného členění, shromažďování informací o potřebném materiálu, tvorbu objednávek a plánování výroby. Tento systém je založený na SŘBD Microsoft SQL Server 2000 a uživatelském rozhraní Microsoft Access 2003.

Jeden z prvních úkolů, které jsem ve firmě řešil, byla analýza a vývoj aplikace, která běží na přenosném počítači PDA se zabudovanou čtečkou čárových kódů. Tato aplikace byla pojmenována SysAdmin Mobile a slouží především pro evidenci polohy skladovaného materiálu.

3.1.1 SysAdmin Mobile

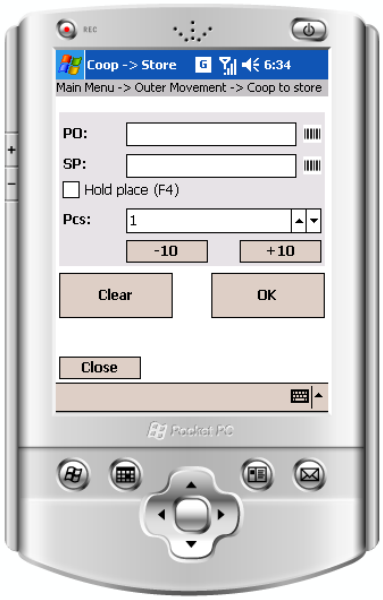
Při vývoji této aplikace jsem se poprvé setkal s programováním pro kapesní počítače PDA s operačním systémem Windows Mobile 5.0 CE. Vzhledem k tomu, že jsem již měl zkušenosti s .NET Frameworkem, nebyl problém zorientovat se ve větvi .NET Compact Framework 2.0, která mi pro vývoj poskytla téměř všechny možnosti jako desktopová verze. Vývoj aplikace trval přibližně 3 měsíce, avšak nové požadavky a úpravy se provádějí neustále.

Rozdílem této aplikace oproti standardním aplikacím nad databází bylo chybějící trvalé připojení do počítačové sítě. Aplikaci bylo nutné přizpůsobit tak, aby se data stahovala a nahrávala na server dávkově při synchronizaci. Vnitřně si tedy program musí pamatovat jednak kopie důležitých dat ze serveru, především číselníky (např. číselníky státních poznávacích značek nákladních automobilů), a dále pak samozřejmě informace, které jsou načteny z čárových kódů mezi synchronizačními body.

Vývoj aplikace byl poměrně náročný na vyladění uživatelského rozhraní, které bylo nutné přizpůsobit jednak malým rozměrům displeje kapesního počítače a jednak provozně-technickým podmínkám ve skladových prostorách firmy. Do vývoje navíc v průběhu vstoupil jiný, levnější model s menším rozlišením displeje, horším kontrastem a bez možnosti dotykového ovládání.

Mobilní počítače, na kterých aplikace běží, jsou vybaveny čtečkou čárových kódů. Tu je možno nastavit tak, aby se naskenovaná data zasílala systému jako běžný vstup z klávesnice, takže nebylo nutné řešit komunikaci aplikace a scanneru.

Účelem je především evidence pozice skladovaného materiálu. Aplikace nabízí funkce přesunu materiálu, který je identifikován tzv. PO-Line (project order – line), což je položka objednávky, která má svůj čárový kód. Aplikace řeší jednak vnitřní pohyb materiálu v rámci skladu, kde existují místa označená jako SP (store places), dále naskladnění, vytvoření a úpravu transportních balíků, tzv. boxů, a přípravu těchto boxů na transport.



Obrázek 1: Emulátor se spuštěnou aplikací SysAdmin Mobile

Veškeré posbírané informace se ukládají do vnitřních paměťových struktur ve formě dávkových příkazů nebo relací. Při synchronizaci, kdy je kapesní počítač napojen na podnikovou síť, se aplikace připojí k SQL serveru a veškerá data nahraje do připravených tabulek. Tyto tabulky jsou odděleny od hlavních datových struktur, aby nedošlo k nechtěnému poškození dat například chybným skenováním údajů ze štítků. Data z těchto tabulek se na straně SQL serveru zpracují a dále slouží jako zdroje pro formuláře hlavní aplikace SysAdmin. Toto finální zpracování již nebylo mým úkolem.

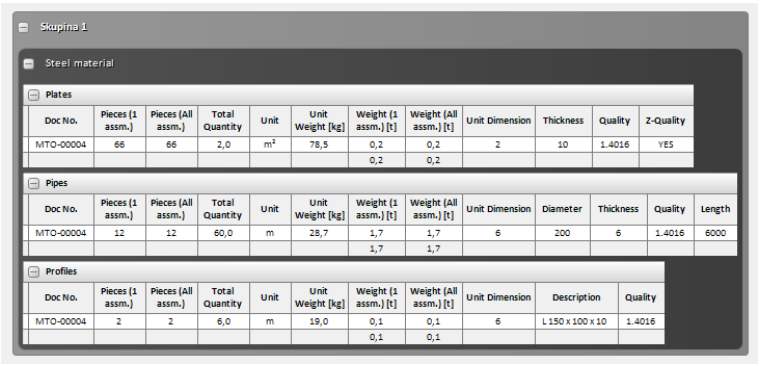
Vzhledem k tomu, že možnosti se průběžně rozvíjí podle potřeb zákazníka, bylo nutné implementovat také mechanismus automatických aktualizací. Soubory nových verzí aplikace ve formátu EXE jsou uloženy v záznamech tabulky verzí jako hodnoty atributu datového typu image (binární data). Při synchronizaci aplikace nahlíží do této tabulky a porovnává číslo své verze s číslem aktuální verze. V případě, že se tato čísla neshodují, nová aplikace se stáhne a uloží do dočasného souboru. Poté se běh SysAdmin Mobile ukončí a spustí se jednoduchá utilita, která nahradí EXE soubor aplikace souborem, který byl stažen. Nakonec proběhne automatické spuštění nové verze.

3.1.2 Material take-off lists

Material Take-off Lists (dále jen MTO) je další informační systém, na jehož vývoji jsem pracoval. V tomto případě jsem se účastnil jak analýzy a návrhu databázového schématu, tak programování samotné aplikace. Vývoj byl zahájen v říjnu roku 2008 a první testovací verze byla uvolněna téhož roku v prosinci. Bohužel je systém stále ve fázi testování a vývoje a průběžně se implementují nové uživatelské funkce. Po nasazení bude MTO sloužit ve výrobním závodě Huisman v České republice a v Nizozemí a po určité době pravděpodobně také v Číně.

Účelem MTO je vytváření soupisů materiálu, který je potřeba objednat na výrobu nějakého projektu. Systém umožňuje zakládání a správu projektů, na kterých se vydefinuje stromová struktura, jež dělí projekty na drobnější celky. Na listové položky tohoto stromu se nakonec navážou standardní díly neboli artikly, které jsou identifikovány pomocí globálně platného čísla – article number. Na rodičovských uzlech listových položek pak vznikají soupisy materiálu, který je potřeba objednat pro vyrobení dané části projektu. Těmto soupisům se říká MTO dokumenty a je nutné evidovat jejich různé verze – revize.

U tohoto systému bylo nutné detailněji analyzovat a vyřešit uživatelská práva. Na editaci projektu má po jeho vytvoření právo pouze zakladatel (supervisor). Ten může definovat další pověřené uživatele (trusted users), kteří pak mají stejná práva jako zakladatel. Na každém uzlu ve stromové struktuře je možné rekurzivně definovat uživatele, kteří budou moci daný uzel dále větvit nebo přidělovat oprávnění. Inspirací při návrhu tohoto mechanizmu mi byl způsob přidělování práv v souborovém systému NTFS.


Skupina 1

Steel material

Plates

Doc No.	Pieces (1 assm.)	Pieces (All assm.)	Total Quantity	Unit	Unit Weight [kg]	Weight (1 assm.) [t]	Weight (All assm.) [t]	Unit Dimension	Thickness	Quality	Z-Quality
MTO-00004	66	66	2,0	m^2	78,5	0,2	0,2	2	10	1.4016	YES
						0,2	0,2				

Pipes

Doc No.	Pieces (1 assm.)	Pieces (All assm.)	Total Quantity	Unit	Unit Weight [kg]	Weight (1 assm.) [t]	Weight (All assm.) [t]	Unit Dimension	Diameter	Thickness	Quality	Length
MTO-00004	12	12	60,0	m	28,7	1,7	1,7	6	200	6	1.4016	6000
						1,7	1,7					

Profiles

Doc No.	Pieces (1 assm.)	Pieces (All assm.)	Total Quantity	Unit	Unit Weight [kg]	Weight (1 assm.) [t]	Weight (All assm.) [t]	Unit Dimension	Description	Quality
MTO-00004	2	2	6,0	m	19,0	0,1	0,1	6	L 150 x 100 x 10	1.4016
						0,1	0,1			

Obrázek 2: Grafická komponenta pro zobrazování MTO dokumentů

Poté, co je na uzlu vytvořen soupis materiálu (MTO dokument), provede se jeho uzamčení a čeká se než vedoucí projektu MTO schválí. Schválením MTO dokumentu vzniká jeho nová revize, odmítnutím se uzel opět odemkne pro úpravy. Revize MTO se v databázi uchovává jako XML dokument, který vznikne zkopírováním záznamů z potomků uzlu, na kterém se revize vytvoří.

Běh databáze systému MTO zajišťuje Microsoft SQL Server 2005, aplikace byla napsána v jazyce Visual Basic 2005 .NET. Jedním z hlavních problémů, které jsou v aplikaci vyřešeny, je stromová organizace projektu. V databázi existuje tabulka položek projektů, která obsahuje vazbu sama na sebe. Touto vazbou, kdy záznam obsahuje odkaz na svého rodiče, je poměrně snadné realizovat strom.

Další problém vyplýval z různých typů dílů – artiklů. Každý artikl má kromě svých základních vlastností, kterými jsou hmotnost, dostupnost na skladě atd., také různé doplňující parametry. U elektromotoru se například eviduje jeho výkon, u plechu je potřeba sledovat kvalitu, tloušťku a jiné charakteristické vlastnosti. Bylo nutné vytvořit přehledné zobrazení MTO dokumentu, kde jsou jednotlivé položky seskupeny do tabulek podle typu navázaného artiklu. Pro vizualizaci MTO dokumentu jsem vytvořil speciální grafickou komponentu, která umí zobrazit a tisknout MTO skládající se z několika různě formátovaných tabulek s různou strukturou. Při tisku je samozřejmě nutné hlídat případy, kdy tabulka přeteče stránku – do šířky nebo do hloubky, situace je zkomplikovaná tím, že každá tabulka obsahuje různý počet sloupců, které jsou různě široké (viz. Obrázek 2).

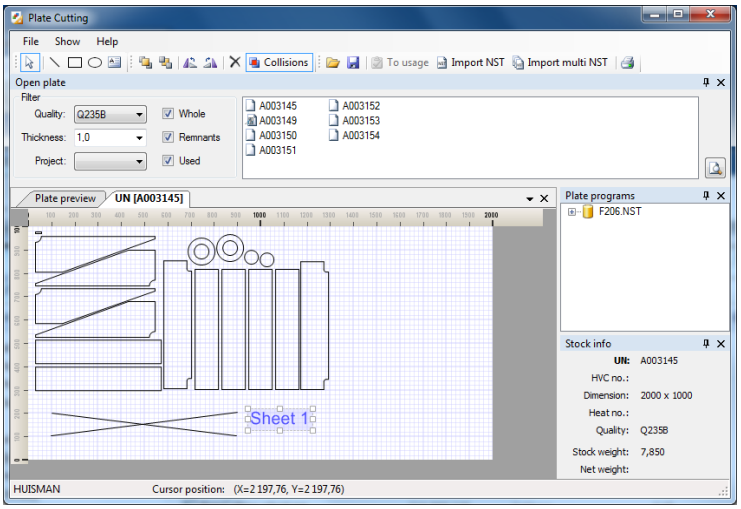
V této chvíli je systém připraven na používání, ale prozatím není jasný způsob, jakým se bude aktualizovat databáze artiklů. Zdrojem bude pravděpodobně skladový systém WMS a aktualizace MTO databáze bude probíhat pomocí DTS balíčků na SQL serveru. I v případě této aplikace bylo

potřeba implementovat mechanismus automatických aktualizací, který je založen na podobném principu jako aktualizace aplikace SysAdmin Mobile.

3.1.3 Plate Cutting

Aplikace Plate Cutting je pravděpodobně nejsložitější problém, který jsem ve firmě z velké části řešil. Jedná se o jednu ze dvou aplikací nově vznikajícího informačního systému, který se bude nasazovat v pobočce Huisman – Fujian v Číně. Tento systém nese souhrnně název FHSysAdmin (Fujian – Huisman SysAdmin) a logika je odvozena z existujícího systému SysAdmin v pobočce zde ve Frýdku-Místku.

Základem jsou podobně jako v případě MTO stromově dělené projekty. Účelem však není vytváření objednávek materiálu, ale naopak registrace spotřebovaného materiálu na projekt. Materiál je v tomto případě rozdělen na plechy, trubky, kulatiny a profily. Trubky, kulatiny a profily se do spotřeby registrují v hlavním modulu FHSysAdmin, registraci spotřeby plechů řeší aplikace Plate Cutting, jejíž vývoj byl mým úkolem.



Obrázek 3: Hlavní okno aplikace Plate Cutting s ukázkou načteného pálícího programu

Složitost této aplikace nebyla v datové komunikaci, ale ve vizualizaci. Spotřeba materiálu je v aplikaci Plate Cutting řešena graficky. S každým záznamem o materiálu typu plech je asociována kresba, která vyjadřuje rozmístění jednotlivých pálících programů na jednom plátu plechu.

Pálící programy jsou tvořeny ve specializovaném software FastCAM, který spolupracuje s CNC stroji pro řezání (vypalování) konkrétních tvarů potřebných součástí. Jednou z hlavních funkcí, které má aplikace Plate Cutting zajišťovat je možnost importu a zobrazení projektových souborů ze systému FastCAM. Jelikož k těmto projektovým souborům neexistuje dokumentace, bylo nutné využít reverzního inženýrství a pochopit jejich strukturu. Naštěstí se jednalo o textové soubory a struktura byla poměrně jasně čitelná. Problémem ale bylo, že se soubor odkazoval na externí DXF výkresy, které definovaly samotné tvary součástí. Existovaly tedy dvě možnosti – zakoupit komponentu pro zobrazování DXF nebo zobrazování DXF naimplementovat.

3.1.3.1 Načítání a zobrazování souborů DXF

Existují dvě varianty DXF – textová a binární. Ve firmě se naštěstí používají pouze textové verze, které jsou exportovány z grafického systému AutoCad 2007. Nakonec bylo rozhodnuto, že zobrazování DXF zajistíme sami, jelikož struktura tohoto poměrně známého typu souboru je detailně zdokumentována.

DXF soubor se skládá z množiny entit, kterými jsou například úsečka, elipsa, kruhová výseč, polygon a podobně. Mimo tyto entity může DXF obsahovat také bloky, což jsou části, které se s různou transformací mohou vyskytovat na různých pozicích. Nebylo nutné implementovat zobrazování všech entit, které dokumentace připouští, například trojrozměrné objekty se na výkresu plechu vyskytovat samozřejmě nemohou.

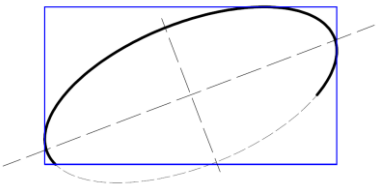
Na základě detailní analýzy možností formátu DXF spočíval první krok implementace ve vytvoření vhodného objektového modelu, který mimo jiné abstraktně definoval, jakým způsobem má vypadat entita. Poté bylo nutné třídy pro jednotlivé entity postupně naimplementovat. Tuto část vývoje jsem přenechal jinému externímu zaměstnanci firmy.

Každá entita musí implementovat metodu pro její načtení z dané části DXF souboru, metody pro serializaci a deserializaci z formátu XML, metodu pro rotaci kolem daného středu o daný úhel a zrcadlení. Mimo to musí být každý objekt entity schopný vrátit množinu bodů, podle kterých je možné vypočítat minimální ohraničující obdélník.

Jakmile bylo možné entity do připravené struktury načíst, bylo nutné provést jejich vizualizaci. Metody pro samotné zobrazení DXF nejsou součástí tříd entit. Celou vizualizaci zajišťuje třída, která pro zobrazování využívá standardní kolekci funkcí .NET Frameworku pro práci s grafikou. Třída DXFImage pak umožňuje vykreslit načtený soubor na zadanou pozici s danou rotací a s daným měřítkem, které může být v ose x i v ose y různé. Celá část aplikace Plate Cutting, která zajišťuje zobrazování DXF souborů byla vyvíjena jako samostatný projekt, jehož výstupem je komponenta – dynamicky linkovaná knihovna. Tuto knihovnu se zdrojovými kódy a ukázkami je možné nalézt na přiloženém CD.

3.1.3.2 Implementace grafického editoru

V momentě, kdy byla komponenta pro zobrazování DXF funkční, přistoupil jsem k programování další části aplikace Plate Cutting, kterou byl vektorový grafický editor. Tento editor měl umožňovat obvyklé základní funkce, jakými jsou kreslení úseček, obdélníků, vkládání textových popisků apod. Všechny vložené entity samozřejmě musí být editovatelné a je možné je odstranit. Editor kromě toho musí být schopný pracovat s kresbou v libovolném měřítku a s různým posunem.



Obrázek 4: Ukázka minimálního ohraničujícího obdélníku kolem eliptického oblouku

Hlavním účelem editoru je ale možnost vkládání kreseb pálících programů ze systému FastCAM, které jsou založeny na souborech ve formátu DXF. Naimportovaný program lze pomocí tažení myši umístit na libovolnou pozici na výkrese. Celá kresba se při ukládání do databáze převádí na formát XML.

Podobně jako při vývoji komponenty pro zobrazování DXF bylo i zde nutné navrhnout objektový model, kde množina tříd popisuje jednotlivé grafické tvary, které se na kresbě plechu mohou nacházet. Množství těchto grafických tvarů je ale podstatně menší než v případě DXF souboru. Naopak každý z tvarů musí implementovat větší množství funkcí, které souvisí zejména s editací. Jedná se například o metody pro změnu tvaru pomocí přesunu uzlových bodů ohraničujícího obdélníku. Dále musí například textový popisek obsahovat zvláštní editor, ve kterém je možné nastavit styl písma

a zarovnání. Obecně má každá entita barvu ohraničení a barvu výplně. Celý DXF soubor, o jehož zobrazování se stará komponenta popsaná výše, je v tomto modelu editoru reprezentován jednou speciální entitou.

Při vývoji grafického editoru a komponenty pro soubory DXF jsem prakticky uplatňoval zejména znalosti analytické geometrie. Například při přetahování tvarů kurzorem myši bylo nutné, většinou pomocí rovnic pro výpočet vzdálenosti bodu od úsečky nebo jiné křivky, stanovit, na jaký objekt kurzor ukazuje. Jedním z největších problémů byl výpočet minimálních ohraničujících obdélníků nad entitami DXF souborů, zejména u tvarů jako jsou oblouky elips, jejichž poloosy nejsou rovnoběžné s osami x nebo y (viz. Obrázek 4). Tento výpočet jsem nakonec prováděl aproximační metodou, kdy jsem nejprve získal množinu bodů daného objektu s parametricky danou hustotou, ze které jsem pak vybíral body s nejmenšími a největšími hodnotami souřadnic.

3.1.3.3 Komunikace aplikace s databází

Veškeré úsilí vývoje této grafické aplikace by samozřejmě nemělo význam, kdyby zde neexistovala komunikace s databází. Kromě toho, že je potřeba uchovávat kresby rozvržení pálících programů na jednotlivých plátech plechů (využívá se atribut záznamu o materiálu typu XML), musí aplikace na základě názvů DXF souborů, které mají ve firmě přesně definovaný tvar, dohledávat položky projektu. Poté, co si pracovníci rozvrhnou, jakým způsobem budou pálící programy rozmístěny, vytisknou připravenou kresbu a provedou samotné vypálení. Po vyřezání označí na kresbě plechu vypálené programy, čímž se vytvoří záznam v párovací tabulce, která eviduje spotřebu mezi položkami projektů a číslem použitého materiálu. U záznamu o tomto materiálu se pomocí databázového triggeru automaticky zmenší jeho množství.

Kromě toho mohou při pálení vzniknout odřezky s poměrně velkou plochou, které lze později využít. S údaji o těchto odřezcích aplikace Plate Cutting také pracuje, mechanismus je však poměrně komplikovaný a zahrnuje i externí skladové informační systémy. Na analýze tohoto procesu, který je nazván jako „odpis materiálu“ jsem se podílel, samotný proces je popsán v projektové dokumentaci FHSysAdmin [3].

3.1.3.4 Úprava aplikace pro běh na pobočce Sviadnov

Aplikace Plate Cutting nebyla zcela novým nápadem, ale koncepčně vycházela ze starší verze, která se již nějakou dobou používá v českém výrobním závodě společnosti Huisman. Na vývoji této původní verze jsem nepracoval, později jsem však implementoval některé funkce, které souvisely s automatickým načítáním plochy při odpisu materiálu. Toto byl jeden z případů, kdy bylo nutné zorientovat se v existujících zdrojových kódech.

Záměr je využívat v české i čínské pobočce novou verzi zejména kvůli stále se rozšiřující stromové struktuře projektů, se kterou původní aplikace nepočítá. Úpravy jsou řízené vnitřní proměnnou o lokalitě, která přepíná dva různé způsoby komunikace s databází. Jedním z hlavních problémů při přizpůsobování Plate Cutting na zdejší podmínky byl rozdílný formát souborů pálících programů a rozdílný proces samotného pálení. Pálící programy jsou načítány ze souborů neoficiálního formátu LOS, jehož struktura je však naštěstí poměrně jednoduchá a celá metoda pro import je součástí zdrojových kódu aplikace Plate Cutting.

3.1.4 Další plánovaná spolupráce

Společným úsilím firem Nativa a Huisman do budoucna je vytvořit globální informační systém – datový sklad HDW (Huisman Data Warehouse), jenž by sjednotil jednotlivé systémy, které fungují prozatím odděleně v různých lokalitách (SysAdmin, FHSysAdmin, MTO a další). Bude se jednat o velmi složitý krok, který se bude rozpadat na mnoho dílčích úkolů. Jedním z nich je sjednocení schémat jednotlivých databází, kde se jedná zejména o jednotný formát stromově dělených projektů. Po technické stránce zde bude nutné využít pravděpodobně replikace dat mezi několika servery, které poběží vždy v každé lokalitě. Situaci mírně komplikuje existence čistě lokálních projektů, které by neměly být zahrnuty do globálního systému HDW.

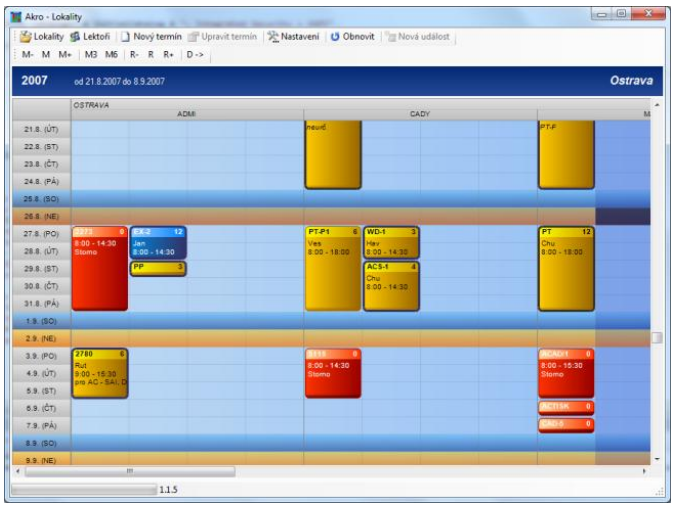
3.2 Školící středisko AutoCont

Dalším zákazníkem firmy Nativa je školící středisko AutoCont v Ostravě. Obě firmy NATIVA Enterprise s.r.o. i AutoCont jsou organizačně úzce spojené, jelikož mimo jiné sdílí společné firemní prostory. Školící středisko firmy AutoCont zaměstnává obvykle poměrně velké množství externích lektorů, kteří vyučují v různých lokalitách.

Ve školícím středisku jsem také mimo jiné vedl několik kurzů, jednalo se převážně o programování pro mládež, avšak tato činnost samozřejmě nespadá do mé odborné praxe ve firmě Nativa.

3.2.1 Grafický rozvrh AKRO

Informační systém pro vedení lektorů, lokalit, vypisování kurzů a podobně, existoval již před mým příchodem do firmy. Situace byla velmi podobná jako v případě systému SysAdmin ve společnosti Huisman. Opět zde existovala a samozřejmě stále existuje databáze běžící na Microsoft SQL Serveru 2000 a hlavní aplikace je naprogramována v prostředí Microsoft Access 2003. Mým úkolem byl vývoj aplikace, která by umožňovala spravovat kurzy v grafické podobě. Aplikace AKRO vznikala souběžně s aplikací SysAdmin Mobile, jednalo se tedy také o jeden s mých prvních projektů ve firmě.



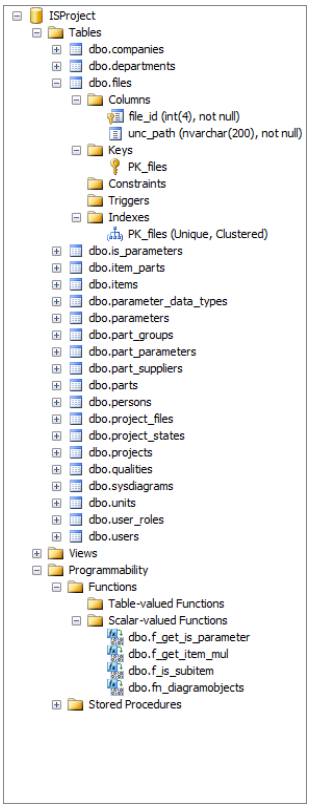
Obrázek 5: Grafický rozvrh AKRO

Grafický rozvrh je tedy v podstatě plánovací systém, kde jsou jednotlivá školení zobrazována v přehledné mřížce organizované podle dnů a lokalit nebo podle dnů a lektorů. Uživatel, který s tímto grafickým rozvrhem manipuluje, může jednoduše zakládat a upravovat kurzy označením příslušných dnů pro potřebnou lokalitu. Kurzy je pak možné upravovat prostým přetažením, kopírovat nebo samozřejmě mazat. Kurzy mají také různé stavy, existují zdvojené kurzy, ve stejný čas může být ve stejné učebně nebo pro stejného lektora naplánováno více školení, zkrátka možností je spousta. Cílem aplikace je tedy co nejvíce usnadnit toto plánování grafickou formou.

Podobně jako v případě Plate Cutting spočívala složitost této aplikace především v grafickém zobrazování. Bylo nutné vytvořit mřížku, jejíž design byl inspirován známým tabulkovým procesorem Microsoft Excel 2003. Zobrazení této mřížky se provádí na základě načtených kurzů, u kterých je evidován seznam dnů, kdy kurzy probíhají, kdo je učí a ve které učebně školení probíhá. Při zobrazování je potřeba identifikovat podle dnů spojité části, aby byl kurz zobrazen jako souvislý obdélník. Začátek znázornění kurzu obsahuje hlavičku se zkratkou názvu kurzu a dalšími údaji jako je počet přihlášených posluchačů nebo cena. Stavy kurzů určují jeho barvu.

Jako implementační prostředí pro tuto aplikaci jsem zvolil platformu Microsoft .NET Framework 2.0. Problematickou částí vývoje této aplikace byla optimalizace rychlosti zobrazování. Kurzů je současně načteno velké množství. Při načítání lze sice vymezit časové období, avšak i přesto může být v jednu chvíli zobrazeno na pracovní ploše hlavního okna aplikace mnoho často jednodenních školení. Poměrně výrazného zlepšení výkonu jsem dosáhl přednačítáním tvarů symbolů kurzů do bitmapových bufferů na pozadí. Při samotném zobrazení se tak nemusí vykreslování rámečků provádět po křivkách znova, ale několikrát se pro stejné typy kurzů použije stejná předkreslená bitmapa. Jeden kurz je složen celkem ze tří bitmap – hlavička, tělo, jehož velikost je proměnlivá, a zápatí.

Tato aplikace se již delší dobu úspěšně využívá a v případě nutnosti jsou dopracovávány požadavky ze strany školícího střediska AutoCont. Při vývoji aplikace jsem se seznámil s prostředím Microsoft .NET Framework 2.0, které momentálně s velkou oblibou využívám.



Obrázek 6: Komponenta pro zobrazení databázového schématu

3.3 Interní projekty firmy

Mimo projekty určené externím zákazníkům jsem vyvíjel také několik menších aplikací, které slouží zejména interním potřebám ve firmě Nativa. Jednou z nich je také nástroj, pro porovnání struktury dvou databází běžících na Microsoft SQL Serveru verze 2005 a výše.

3.3.1 SQL Compare

Účelem aplikace je porovnávat a zjišťovat změny mezi dvěma databázovými schématy. Obvykle existuje jedna verze databáze, která funguje u zákazníka a jedna verze, která slouží firmě pro vývoj. Ve vývojové verzi se provádí nejrůznější úpravy, tím je myšleno přidávání a odebírání sloupců v tabulkách, vytváření tabulek nebo pohledů, vytváření a úprava funkcí, uložených procedur nebo triggerů, tvorba indexů a mnoho dalších operací. Před tím, než je uvolněna nová verze aplikace, je nutné přenést všechny změny, které se provedly na databázovém schématu při vývoji, do databáze zákazníka. Pro zjištění a následné provedení těchto změn slouží právě aplikace SQL Compare.

Aplikace je zatím stále ve vývoji, avšak poslední verze již dokáže spolehlivě rozpoznat všechny důležité změny. Jako implementační prostředí jsem zvolil Microsoft .NET Framework 3.5, na kterém jsem si prakticky vyzkoušel technologii LINQ.

Před samotným porovnáním dvou databázových schémat se obě schémata načtou do aplikace. Pro uchovávání a následné porovnání bylo nutné vytvořit objektový model, kde je pro každý typ objektu v databázi implementována třída v aplikaci. Tento model vychází z principů, které definují pohledy ve schématu SQL serveru „.sys“ (sys.objects, sys.trigger_events atd.). Pomocí údajů z relací v tomto schématu se také provede hromadné načtení všech objektů dané databáze.

Pro zobrazení stavu databázového schématu jsem vytvořil komponentu, která napodobuje funkčnost stromového zobrazení objektů v SQL Management Studiu 2005 (viz. Obrázek 6).

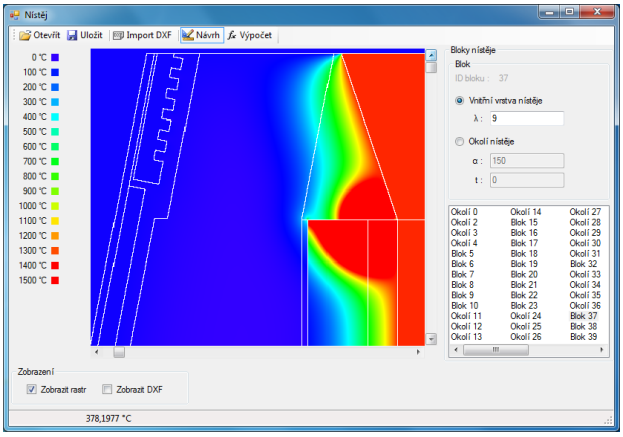
Jakmile jsou obě schémata načtena a vnitřně reprezentována objektovým modelem, provede se samotné porovnání. Právě zde jsem využil možností technologie LINQ, zejména při provádění doplňku, průniku nebo rozdílu množin stejných typů objektů mezi oběma schématy. Zjišťoval jsem tak například odstraněné, upravené nebo nově vytvořené tabulky.

Také pro zjištěné změny bylo nutné vytvořit objektový model, jehož struktura bere v úvahu jak typy změn podle typu měněného objektu (trigger, tabulka, pohled), tak typy změn podle smyslu (create, alter, drop). Hierarchie tříd vychází z typů měněných objektů, smysl změny se do modelu projevuje implementací tří různých rozhraní.

Vývojová verze aplikace se nachází na přiloženém CD. V současné době aplikace umí velmi efektivně a rychle porovnat dvě schémata a v blízké době bude implementována také možnost generování SQL skriptů, pomocí kterých se bude cílová databáze upravovat.

3.4 Další projekty

Uvedený výčet prací zdaleka není vším, na čem jsem se při vývoji nebo analýze podílel. Poměrně důležitým zákazníkem, který zde nebyl uveden, je společnost Mölnlycke Health Care, jejíž výrobní závod sídlí v Karviné. Pro tuto společnost jsem se podílel na vývoji systému pro tisk etiket – štítků, které označují vyrobené zboží. Zajímavou zkušeností byla zejména manipulace s čárovými kódy typu EAN 128 C, které jsem vlastním algoritmem sestavoval.



Obrázek 7: Vývojová verze aplikace pro výpočet rozložení teplot v nístěji

V současné době pracuji na speciálním výpočetním software pro společnost BKB Metal, který bude sloužit při výpočtu rozložení teplot v nístěji pro tavení železa. Nístěj se skládá z několika bloků, které mají různé vlastnosti a teplo se jimi šíří jiným způsobem. Pro import kresby řezu nístějí jsem využil již existující komponentu pro načítání DXF souborů. Kresba DXF souboru se po načtení převede na rastr, ve kterém je dle způsobu ohraničení každé buňce stanoven vzorec, podle kterého se v ní spočítá teplota. Záleží tedy, zda buňka leží na rozhraní dvou nebo více bloků nebo se jedná o buňku uvnitř bloku.

Častou pracovní náplní byla také implementace mechanismů, které převádějí data z jedné databáze do druhé. Tyto mechanismy obvykle pracují na základě transakčních balíčků DTS, jejichž funkčnost je vnitřně daná posloupností serverových operací nebo skriptů. Jednalo se jak o přenos dat mezi dvěma databázemi na SQL serveru, tak například o import dat ze souboru ve formátu XLS. Pomocí těchto metod se obvykle plní databáze například informacemi o dostupném materiálu.

Jak zde bylo již několikrát zmíněno, kromě samotného programování jsem se průběžně podílel na analýze. V této souvislosti bylo třeba konzultovat a vyhodnocovat požadavky zákazníků, nebo naopak navrhovat, jakým způsobem by bylo možné aplikaci rozšířit. Zde jsem se snažil využívat znalosti z oblasti softwarového inženýrství například využíváním UML diagramů.

4 Uplatnění znalostí a dovedností získaných v průběhu studia

V této kapitole bych rád shrnul veškeré teoretické a praktické znalosti, které jsem nabyl v průběhu studia a uplatnil je při výkonu odborné praxe. Nabyté znalosti mi nejen pomáhaly při řešení samotných problémů, ale v mnoha případech mě inspirovaly k návrhu dalšího postupu vývoje nebo analýzy různých systémů nebo aplikací. Obvykle však platilo, že okruh znalostí, který jsem zamýšlel pro řešení daného problému využít, bylo potřeba v souvislosti s tímto problémem detailněji nastudovat. Vzhledem k tomu, že ve firmě působím již delší dobu, pomáhala mi často naopak praxe k lepšímu pochopení významu některých teoretických znalostí, kterých jsem nabýval studiem později.

Dovednosti spojené se samotným programováním jsem získával převážně samostudiem, avšak předměty jako Programovací techniky nebo Základy algoritmizace mi nejednou ukázaly možnosti programovacích jazyků, o kterých jsem ani netušil, že existují. Mezi ně patří například využívání generických datových typů a kolekcí, které mi výrazným způsobem usnadnily vývoj. Dále také pochopení datových struktur, jakými jsou například hash tabulka nebo přímé adresování, mě inspirovalo při optimalizaci výkonu některých aplikací (např. grafického rozvrhu AKRO).

Dobrá znalost jazyka SQL pro dotazování nad databází je v případě implementace většiny informačních systémů nutná. Zde bych chtěl vyzdvihnout především předmět Databázové systémy, který jsem absolvoval ve třetím ročníku na střední škole SPŠ Kratochvílova a který mi poskytnul většinu potřebných teoretických i praktických znalostí jazyka SQL. V souvislosti se studiem na vysoké škole jsem důležité informace ohledně principů fungování transakcí získal také předmětem Databázové a informační systémy.

V oblasti analýzy informačních systémů jsem velmi často čerpal z předmětu Softwarové inženýrství, kde mě velmi zaujala myšlenka vizuálního jazyka UML. Komunikace analytika a zákazníka je při vývoji informačního systému jedním z největších problémů. Úkolem analytika je pochopit, co vlastně zákazník potřebuje a naopak zákazník si potřebuje vytvořit představu o tom, jak bude navrhovaný informační systém fungovat a vypadat. Zde jsem se při dokumentaci procesů snažil využívat UML a vývojové diagramy, např. při návrhu importu materiálu ze skladového systému WMS.

Z mého pohledu jedním z nejzajímavějších úkolů byla tvorba komponenty pro zobrazování DXF souborů. Jak jsem již uváděl, zde bych se neobešel bez znalostí analytické geometrie, se kterou se většina lidí setká jen teoreticky.

5 Znalosti, které jsem postrádal

Vědomosti v oblasti samotného programování zejména na platformě .NET jsem získával především formou samostudia. Řešení většiny problémů vyžadovalo vlastní iniciativu z hlediska nabývání nových schopností. Bylo nutné studovat různé speciální možnosti využívání jak jazyků pro vývoj aplikací (Visual Basic .NET nebo C#), tak jazyka T-SQL na straně Microsoft SQL Serveru. Jednalo se například o tvorbu komponent, dynamicky linkovaných knihoven, instalátorů nebo transakčních

balíčků. V souvislosti s jazykem C# jsem využíval především knihu Visual C# 2005 [1] a online knihovnu MSDN Library [2].

Vzhledem k tomu, že jsem se angličtině věnoval již v minulosti, netýká se tento problém přímo mě, ale pokud bych měl možnost zdůraznit nějaký okruh, kterému by, dle mého názoru, měla škola věnovat větší pozornost, pak se bude jednat zejména právě o výuku technicky a prakticky zaměřeného cizího jazyka – angličtiny. Její znalost je v současné době velmi důležitá, jelikož umožňuje rozšířit pole působnosti i do zahraničí. Ať se jedná o tvorbu nápovědy, komunikaci se zákazníky obvykle formou e-mailu nebo o volbu vhodných identifikátorů ve zdrojovém kódu, ve všech těchto případech je aktivní znalost anglického jazyka téměř nutností.

6 Závěr

Teorie se od praxe vždy trochu liší. Odlišnosti vznikají podle mě hlavně tím, že vyvíjený software musí odpovídat především požadavkům zákazníka. Mezi tyto požadavky spadají, kromě nároků na samotnou funkčnost, také nároky na cenu a termín. Bohužel urychlování vývoje a snižování nákladů se negativně projevuje na kvalitě, čímž trpí v dnešní době velké množství komerčního software.

Styl psaní zdrojového kódu se u každého programátora průběžně vyvíjí. Nikdy není možné napsat dokonalý software, který funguje od data vydání bezchybně. Vyladění aplikace nebo databázového schématu vyžaduje spolupráci s cílovým uživatelem a jedině průběžným testováním je možné chyby odhalit. Po ukončení vývojového cyklu je vhodné zamyslet se nad částmi kódu, které by bylo možné napsat efektivnějším způsobem a ze zjištěných nedostatků se do dalšího vývoje poučit.

Vykonávání odborné praxe ve firmě NATIVA Enterprise s.r.o. mi přineslo velké množství praktických dovedností a budu velice rád, pokud mi firma umožní i další spolupráci na mnoha zajímavých projektech, díky kterým budu moci nových zkušeností dále nabývat.

Seznam použité literatury

[1] SHARP, John. *Microsoft Visual C# 2005 : krok za krokem*. Brno : Computer Press, a.s., 2006. 528 s. ISBN 80-251-1156-3.

[2] *Microsoft Developer Network* [online]. 2010 [cit. 2010-04-18]. Dostupné z WWW: <http://msdn.microsoft.com>.

[3] *Projektová dokumentace firmy Nativa*. Dostupné pouze pro interní potřeby firmy.

[4] *Huisman* [online]. 2008 [cit. 2010-04-19]. O společnosti Huisman. Dostupné z WWW: <http://www.huismanequipment.com/cz/about_huisman>.

[5] *NATIVA Enterprise s.r.o.* [online]. 2010 [cit. 2010-04-28]. Dostupné z WWW: <http://www.nativa.cz/>.

Seznam obrázků

Přílohy

I. CD Bakalářská práce